IN THE CLAIMS

Please cancel claims 19-25 without prejudice or disclaimer.

This listing of claims will replace all prior versions, and listings, of the claims in the Application.

Listing of Claims:

Claim 1 (original) A method for implementing a polymorphic call site in a computer system executing an object oriented program, said method comprising the steps of:

a) creating a template of a polymorphic inline cache for a polymorphic call site, the template having a plurality of slots;

b) executing the polymorphic inline cache with an object of type k;

c) invoking a polymorphic inline cache initialisation routine;

d) finding an available k slot of the polymorphic inline cache;

e) locking the k slot of the polymorphic inline cache;

f) searching for a k method to call for the object of type k;

g) filling the k slot with a call instruction to the k method;

h) unlocking the k slot to complete the k slot; and

i) calling the k method of the object of type k.

Claim 2 (original) The method of claim 1, wherein a first thread may initialize and/or access the k slot at the same time a second thread is initializing and/or accessing a slot other than the k slot.

Claim 3 (original) The method of claim 2, wherein a second thread calling the k method while the first thread is initializing the k slot waits until after said unlocking step (h).

Claim 4 (original) The method of claim 2, wherein a second thread calling the k method while the first thread is initializing the k slot searches for and calls the k method and leaves the polymorphic inline cache unchanged.

Claim 5 (original) The method of claim 1, wherein a polymorphic inline cache is created for each polymorphic call site.

Claim 6 (original) The method of claim 1, wherein said creating step (a) further comprises inserting a first illegal type value in a compare instruction of every slot of the polymorphic inline cache to indicate each slot is empty.

Claim 7 (original) The method of claim 1, wherein said creating step (a) site further comprises putting a bit in each slot to indicate that each slot is empty.

Claim 8 (original) The method of claim 6, wherein said locking step (e) further comprises replacing the first illegal type value in the compare instruction of the k slot with a second illegal type value to indicate the k slot is in use.

Claim 9 (original) The method of claim 1, wherein said locking step (e) further comprises changing a bit in the k slot to indicate that the k slot is in use.

Claim 10 (original) The method of claim 8, wherein said unlocking step (h) further comprises replacing the second illegal type value in the compare instruction of the k slot with a value of type k.

Claim 11 (original) The method of claim 1, further comprising the step of (j) updating the polymorphic inline cache so that an object of type k+1 will initialize a corresponding k+1 slot.

Claim 12 (original) The method of claim 11, wherein said updating step (j) further comprises inspecting the polymorphic inline cache to find the next empty slot.

Claim 13 (original) The method of claim 12, wherein said polymorphic inline cache initialisation routine is the same for every object of the polymorphic inline cache.

Claim 14 (original) The method of claim 11, wherein said updating step (j) further comprises maintaining a state of the k slot or the k+1 slot.

Claim 15 (original) The method of claim 14, wherein said polymorphic inline cache initialization routine is the same for every object of the polymorphic inline cache.

Claim 16 (original) The method of claim 15, wherein said updating step (j) further comprises modifying the state of the k+1 slot.

Claim 17 (original) The method of claim 1, wherein said invoking step (c) further comprises calling a different initialization routine for every object of a different type.

Claim 18 (original) The method of claim 11, wherein said updating step (j) further comprises modifying the initialization routine so that upon a cache miss of the k slot, the k+1 initialization routine is called.

Claims 19-25 (cancelled)

Claim 26 (original) A computer system for executing an object oriented program, comprising:

means for executing an object oriented program;

means for calling a first method from a first slot of a polymorphic inline cache;

means for calling a second method from a second slot of the polymorphic inline cache;

means for determining if the first method and the second method have an identical object type;

means for calling the first method and the second method simultaneously if they do not have the identical object type; and

means for preventing calling the second method from the second slot of the polymorphic inline cache until said means for calling the first method has completed if they have the identical object type.

Claim 27 (original) The computer system of claim 26, wherein the first method may be called from a first thread and the second method may be called from an independently executing second thread.

Claim 28 (original) The computer system of claim 27, further comprising:

means for invoking a first polymorphic inline cache initialisation routine;

means for locking the first slot of the polymorphic inline cache;

means for filling the first slot with a call instruction to the first method while the first slot is locked;

means for updating the polymorphic inline cache so that a second method of a second type will invoke a second polymorphic inline cache initialisation routine;

means for making the first slot available to the first method;

means for calling the first method;

means for locking the second slot of the polymorphic inline cache;

means for filling the second slot with a call instruction to the second method while the second slot is locked; and

means for updating the polymorphic inline cache so that a Nth method of a Nth type will invoke a Nth polymorphic inline cache initialization routine.

Claim 29 (original) The method of claim 28, wherein the first, second, and Nth polymorphic inline cache initialization routines are identical.

Claim 30 (original) The method of claim 28, wherein the first polymorphic inline cache initialisation routine is called from a first thread simultaneously while an independently executing second or Nth thread is calling the second or Nth method or is calling the second or Nth polymorphic inline cache initialization routine.